TicketBoss

Group Assignment

Design the Architecture for the following system:

TicketBoss is a system to handle ticket sales for various entertainment events. It does the following:

- It displays multiple entertainment events that a user can purchase tickets for
- When a user selects an event, it shows seats available. A user can select seats, and see prices for those seats.
- When a user selects seats, the seats are held for ten minutes. If the user does not complete the purchase within ten minutes, the seats are released for someone else to buy.
- Purchasing tickets involves the usual stuff: user enters credit card information, billing address, etc. TicketBoss emails a confirmation to the user. (Design note: there are applications available that handle credit card payments.)
- Multiple users can access the same entertainment event at the same time, but of course, only one person can hold a particular seat.
- A user can hold up to 10 seats at a time.

Do the following:

- Design the architecture.
- Document the architecture. You should include the following:
    - Logical view: Components and connectors diagram
    - Process view: Sequence diagrams: interaction among components
    - Use-case view: (NOT use case map; don't do all the use cases – fully write use cases 1 and 4 – UC4 will be kind of tricky because you have two primary actors.) (see details about documenting use cases here)
    - Note: The physical view should be pretty boring: just a client-server, or broker, etc. You may include one, but that's not where the interesting stuff is.
- Provide additional textual documentation to give the big picture and explain how everything will work. Include a general explanation of how everything hangs together. You might also include:
    - Discussions of important quality attribute requirements, and how they will be addressed.
    - Describe the architecture patterns used (it should be clear in the diagrams, but sometimes additional text is needed as explanation.)
    - Describe why you made certain key architectural decisions (rationale).

It is recommended that you do this ~~in pairs~~ altogether in the group (~~exactly two people.~~ all members). Do all your design work together in front of a white board.

Use Cases:

1. User selects seats and buys them
2. User selects seats and allows them to expire
3. User selects seats, and then before the first ones expire, selects other seats for the same event. (Hmm. What should the system do?)
4. Two or more people select seats for the same event at the same time (within ten minutes of each other.) There is no conflict.
5. Two or more people select seats for the same event, with a conflict (they try to select the same seats.)
6. User views seats without selecting any
7. User is a new user
8. User holds seats in two or more events at a time (How does checkout work?)
9. User tries to exceed the maximum by holding multiple seats in multiple events.
10. User holds multiple seats to the max. Some expire, and the user gets more seats, reaching the max again.

Plan to do a 10 min presentation in class.